Bioconductor Developers' Forum
Sep 19, 2019

DataFrame/DFrame

Hervé Pagès
Core Team

hpages@fredhutch.org

Concrete class vs virtual class

Concrete class: can be instanciated

```
> setClass("C", slots=c(stuff="ANY"))
> c <- new("C")
```

Virtual class: cannot be instanciated

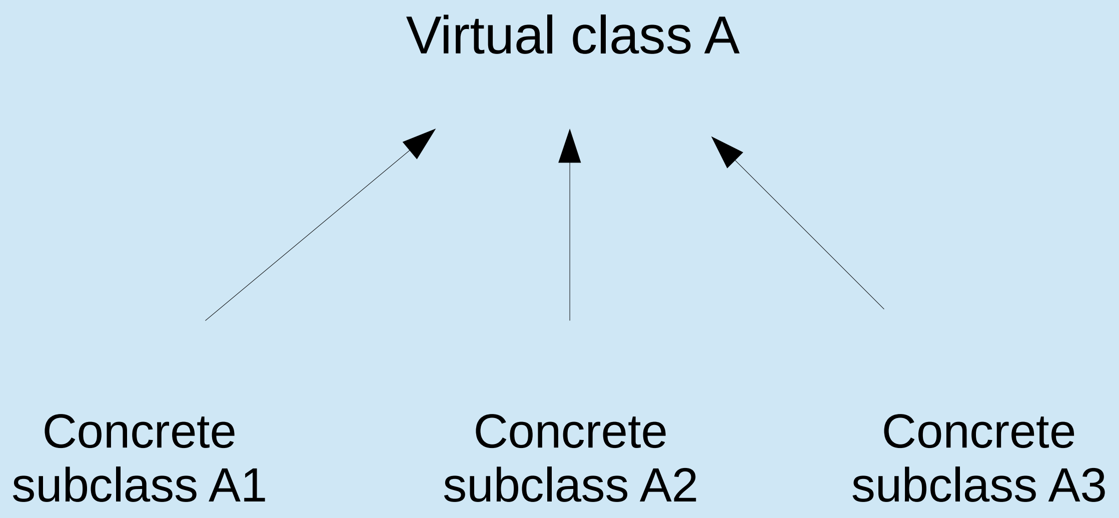
```
> setClass("V", contains="VIRTUAL", slots=c(stuff="ANY"))
> v <- new("V")
Error in new("V") :
  trying to generate an object from a virtual class ("V")
```

When to use a virtual class

Typically when we want to support more than one implementation of the same “concept”.

The virtual class represents the concept.

Each concrete subclass implements the concept.



Examples of virtual classes in Bioconductor

$$\begin{tikzcd}
 & \texttt{CharacterList} & \\
\texttt{SimpleCharacterList} \arrow[ur] & & \texttt{CompressedCharacterList} \arrow[ul]
\end{tikzcd}$$

```
> x1 <- CharacterList(letters[1:5], c(NA, "foo"))
> x1
CharacterList of length 2
[[1]] a b c d e
[[2]] <NA> foo
> class(x1)
[1] "CompressedCharacterList"
attr(,"package")
[1] "Iranges"

> x2 <- CharacterList(letters[1:5], c(NA, "foo"), compress=FALSE)
> x2
CharacterList of length 2
[[1]] a b c d e
[[2]] <NA> foo
> class(x2)
[1] "SimpleCharacterList"
attr(,"package")
[1] "IRanges"
```

2 different CharacterList implementations

$$\begin{tikzcd} & \texttt{CharacterList} & \\ \texttt{SimpleCharacterList} \arrow[ur] & & \texttt{CompressedCharacterList} \arrow[ul] \end{tikzcd}$$

```
> getSlots("SimpleCharacterList")
      elementType     elementMetadata            metadata            listData 
      "character" "DataTable_OR_NULL"              "list"              "list" 


> getSlots("CompressedCharacterList")
      elementType     elementMetadata            metadata          unlistData 
      "character" "DataTable_OR_NULL"              "list"               "ANY" 
     partitioning 
"PartitioningByEnd" 
```

Data-frame-like containers

DataFrame

```
> getSlots("DataFrame")
           rownames                nrows             listData          elementType
"character_OR_NULL"            "integer"               "list"          "character"
    elementMetadata             metadata
"DataTable_OR_NULL"               "list"
```

- The columns are stored in an ordinary list (`listData` slot).

SQLDataFrame (work-in-progress)

- Qian Liu, https://github.com/Bioconductor/SQLDataFrame

- The data is in an SQL database

- Ideally, we’d like to be able to use an SQLDataFrame object everywhere a DataFrame is expected e.g. for the metadata columns of a Vector derivative:

```
> mcols(x) <- SQLDataFrame(…)  # should work
```

The virtual class approach

$$\begin{tikzcd}
 & & \texttt{DataFrame} & & \\
\texttt{DFrame} \arrow[urr] & \texttt{SQLDataFrame} \arrow[ur] & & \texttt{ParquetDataFrame} \arrow[ul] & \texttt{...} \arrow[ull]
\end{tikzcd}$$

- `DataFrame` becomes a virtual class with no slots

- `DFrame` replaces the original `DataFrame` (in-memory representation)

- Other `DataFrame` derivatives implement on-disk representations

- They all support the DataFrame API (`dim`, `nrow`, `ncol`, `[`, `[[`, etc...) possibly with some restrictions (e.g. `[[<-`) for some of the derivatives

- They all should be usable where a DataFrame is expected

- `as(DF, "DFrame")` becomes the standard idiom to load the data in memory

Other benefits

Some methods can be defined at the level of the virtual class:

```
setMethod("dim", "DataFrame", function(x) c(nrow(x), ncol(x)))

setMethod("length", "DataFrame", function(x) ncol(x))

setMethod("dimnames", "DataFrame", function(x) c(rownames(x), colnames(x)))

etc...
```

–> avoids code duplication across the `DataFrame` derivatives.

Work in progress

- `DFrame` added to the devel version of S4Vectors in August

- `DataFrame()` constructor and `as(x, “DataFrame”)` now return an object of class `DFrame`

- `DFrame` objects are displayed as being DataFrame object (they are, `is()`)

- Change is transparent for the end user

- Mostly transparent to the developers (use `is(x, "DataFrame")` rather than `class(x) == "DataFrame"`)

- There is more to complete the `DataFrame`-to-`DFrame` migration

Thanks!

Questions?